A Fully Scalable and Distributed Architecture for Video-on-Demand

Fernando Cores, Ana Ripoll, Emilio Luque

Computer Science Department - University Autonoma of Barcelona – Spain
Fernando.Cores@uab.es A.Ripoll@cc.uab.es E.Luque@cc.uab.es

Abstract. In spite of the attractiveness of Video-on-demand (VoD) services, their implantation to the present has not been as widespread as could have been desired due to centralized VoD systems have a limited streaming capacity and its grown is costly. One level proxy-based systems have been proposed to increase the system capacity but their scalability are still limited by the main network bandwidth. Our investigation are focussed on designing a flexible LVoD (large-scale Video-on-Demand) system capable of easy scaling with limited costs, which can adapt its size to the needs of the system. To achieve a scalable LVoD system, it is essential that the communications system bandwidth is able to grow in keeping with system growth (with a reasonable cost and limited loss of efficiency). To get these requirements we have proposed a hierarchical tree topology based on the use of independent local networks with proxies. To allow the system's grow, the functionality of the proxy has been modified in such a way that it works at the same time as cache for the most watched movies, and as a mirror for the remaining movies. The evaluation of these systems has been done using an analytical model. The results shows that this architecture guarantees unlimited and low-cost growth for LVoD systems, the VoD system capacity can easily be adapted to any number of users and the system is fault-tolerant.

1 Introduction.

Video-on-demand (VoD) systems have been one of the most active areas of research in recent years due to the coming together of two factors: the growing interest from diverse sectors of industry (entertainment, education, information, cable companies, telecommunications, etc.) in developing such systems, and the great complexity that these systems have (integrating various types of information together with real-time requisites, very strict quality of service levels, a great number of users and a high volume of information).

In spite of the attractiveness of these services for the public in general, their implantation to the present has not been as widespread as could have been desired. The construction of large-scale video-on-demand (LVoD) is currently limited both by the capacity of the server as well as by the capacity for simultaneous transmissions that can be supported by a communication network (network bandwidth), limitations on the subsequent growth of these systems, the significant initial investment required and high maintenance costs, among others.

For the large-scale distribution of multimedia content in wide-area networks for commercial use, requirements are not fulfilled by any of the currently existing approaches. These systems would require arranging the servers that provide video retrieval and playback services in a distributed system in order for them to support a large number of concurrent streams. Furthermore, the construction of these systems would require substantial initial investment. This is due to the need to create larger systems than those originally required, given their zero scalability.

LVoD systems would be scalable, because of the need for their rapid expansion, caused by an increase in users or by the inclusion of new services (video conferences, Internet, etc.). Failure to contemplate scalability in the design of such systems may lead to system overload, which in turn could cause client service rejection, since the requirements of real-time streaming for video playing are not to be guaranteed.

In order to achieve scalable LVoD, it is essential that the communications system bandwidth is able to grow in keeping with system growth. It is also essential that this growth be produced at a reasonable cost and with limited loss of efficiency. The aim of our research is to obtain a LVoD system in which system size does not depend on currently-available technology, and that can adapt itself to requirements of system-user growth. In this paper, we focus on the following objective: achieving a flexible LVoD system capable of easy scaling, with limited costs.

The paper is organized in the following way: in section 2, we will first undertake an overview of the solutions proposed in the literature for increasing VoD system capacity. Following this, in sections 3 and 4, we will study the factors that limit scalability in the current systems, and will outline our proposal based on a hierarchical topology. In order to evaluate these systems, in section 5, we will propose an analytical model, and in section 6, will study the efficiency and scalability of the systems in question. Finally, in the last section, we will indicate the main conclusions to be drawn from our discussion, and we will suggest lines of future research deriving from this study.

2 Related Work.

In past years, research into VoD systems has mainly been focussed on policies that attempt to improve the available bandwidth efficiency. These techniques basically aim at increasing the number of users that can be served with a limited bandwidth. Such approaches are grouped into two broad policy groups: broadcasting and multicasting techniques.

In broadcasting techniques, streams (minimum unit of video transmission) are sent to all clients, and they subsequently decide whether the information interests them. If they are not interested, they can simply ignore it. There are various broadcasting techniques, and these are differentiated principally in terms of the number of streams used in the broadcasting of a movie and the transmission frequency of each one of the pieces into which the video is divided. Some of the most widely-used broadcasting techniques are pyramid [11] and skyscraper broadcasting [6].

The main difference between broadcasting and multicasting techniques is the range of clients to whom the streams are directed. With multicasting techniques, information is sent to a reduced group of clients who have previously requested the data received, and as a result, there is never any bandwidth wastage. The most widely-

known multicasting techniques are batching [1][2], piggybacking [3], patching [7][9] and merging [4][13].

All of these techniques aim to improve the bandwidth performance available within the system, but do not increase it. One of the solutions proposed for increasing the size of the VoD systems is the connection to users via independent networks [5]. In these systems, users are grouped in networks segments known as local networks, whose traffic is independent with respect to other segments, in such a way that the system bandwidth is able to be, at least in theory, the bandwidth accumulated by each one of the individual networks. Nevertheless, in order to increase the bandwidth in these systems, it is not enough simply to group users into independent networks, since, if all users have to access the same server and its network, this would create a bottleneck, and the system would therefore be saturated.

The key to the successful of these systems with independent networks and to obtaining better performance lies in the fact that certain requests can be served locally without the need to access the origin of the data through the main network. Various techniques are proposed in the literature to realize this objective. Some are based on placing VoD servers close to clients' networks so that these users are not required to access the server, and thereby creating a system of hierarchical servers [12]. This strategy involves considerable cost; as a result, certain proposals have opted for reducing the size of local servers in such a way that they do not store all the movies available within the system, but instead, only those with a higher access frequency. These servers are managed as main server caches and are called proxies, just as their Internet counterparts. There are a variety of policies for managing proxy content, depending on whether they store complete movie or only certain parts (prefix-proxy[8][14]).

The general topology of a system based on proxies (see Fig. 1) consists of a main server, which is connected, through the main network, to a group of independent networks with their proxy.

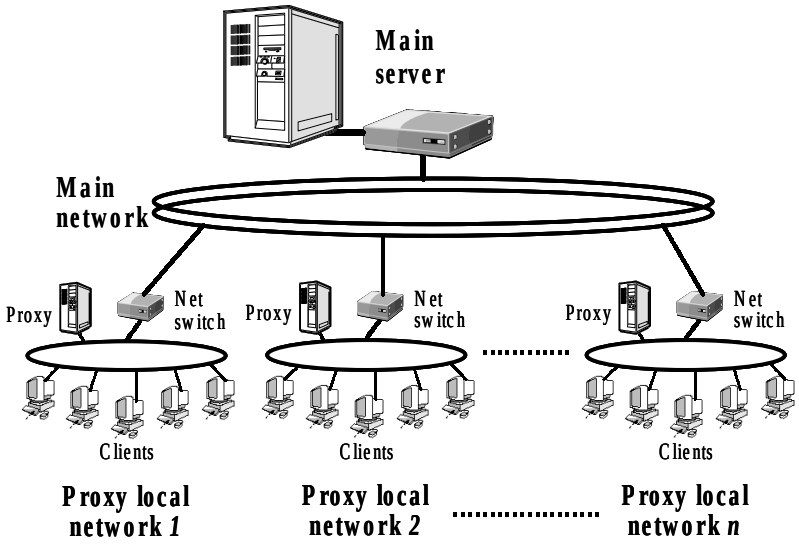

Fig. 1. One-level proxy VoD system.

Another alternative proposed for increasing the size of VoD systems is the distribution of request management throughout all the system components, allowing the clients themselves to serve other users, through their local buffer (Chaining[10]).

3 Scalability in VoD Systems.

Of the main components of a VoD system (server and transmission network), server bandwidth is always greatest, given that bus technology offers a better cost/performance ratio and better scalability. This scalability can be achieved through the inclusion of new disks, using cluster methodology or using various independent servers connected to the same service network. Even with this, however, the establishment of VoD servers capable of handling thousands of simultaneous streams is both complex and costly, due to the high performance level required.

On the other hand, network bandwidth is smaller (due to its associated costs), becoming the true bottleneck when the system grows, and considerably limiting the system's final size. The reason for its poor scalability is that network bandwidth is limited by currently-available technology and it can not grown. The system network bandwidth available only can be increased either by the inclusion of independent servers with their own service networks (implying high costs due to the need for a new server, and a poor sharing of streams between clients) or by the inclusion of additional networks to the system server (implying that growth is limited by installation type, that a more complex server is necessary and that there will be a poor stream sharing). In short, the final size and growth of VoD systems depend, ultimately, on the capacity (bandwidth) of the connection network with the end client.

As we have already seen, the solution proposed by the literature to the problem of increasing service capacity in VoD systems is the use of a central server that is connected, via the main network, to a group of independent networks with their proxy. The effectiveness of these proxies is based on the pattern of user requests for movies following a Zipf(0) distribution [1], few movies have a high percentage of total requests, and therefore, even with a small cache, a high percentage of requests can be attended (e.g. with a cache of 20% of movies available, values in the region of 50% success can be attained). This percentage of success is not sufficiently high to avoid server congestion if there is an excessive increase in the number of local networks. The problem with these systems is that requests that cannot be served locally end up in the main server, which becomes, once again, both a bottleneck and a growth-limiting factor. In spite of these systems obtain greater system's effective bandwidth in comparison to centralized systems without proxies, they do not successfully solve either the problem of the system's limited size or that of scalability. Since, as clients number grow (with the consequent increase in local networks), there comes a point at which the network and main server cannot cope with the traffic generated by proxy cache misses, and its growth is halted. Consequently, there is the renewed need for entire system replication techniques, or for a re-dimensioning of the server and main network in order to increase system capacity.

In order to attain a scalable communication system, it is essential to solve the (physical) limitations of centralized systems (a single network or single server), as these will always be constrained by currently-available technology. For this reason,

we will focus on distributed systems in order to facilitate future system growth. The distributed topology proposed is described in the following section.

4 Our Approach: a Hierarchical Proxy VoD System.

Our proposal consists of an expansion of the proxy technique, currently restricted to a single level, using a tree topology that provides the system with unlimited scaling capacity as well as greater flexibility when deciding on its size and its form of growth.

The structure of this topology, shown in figure 2, consists of a series of levels, in accordance with the number of local networks and the order of the tree (binary, tertiary, etc.). All the topology nodes have the same components: a network, a proxy and its clients.

The main network, to which both the VoD server as well as the first level hierarchy is connected, is located on the main level. The first level hierarchy is made up of a series of local networks (depending on the tree order) that form the following tree level (level 1). Subsequent networks are successively hung from each one of the previous local level networks until reaching the final level. It must be emphasized that this topology requires neither that the levels be complete (although it is recommendable to fill a given level before starting on the next, in order for the system to be balanced), nor that all the tree levels have the same number of connected networks (the same order). Furthermore, if the main network has also clients connected, we then have a homogeneous system.

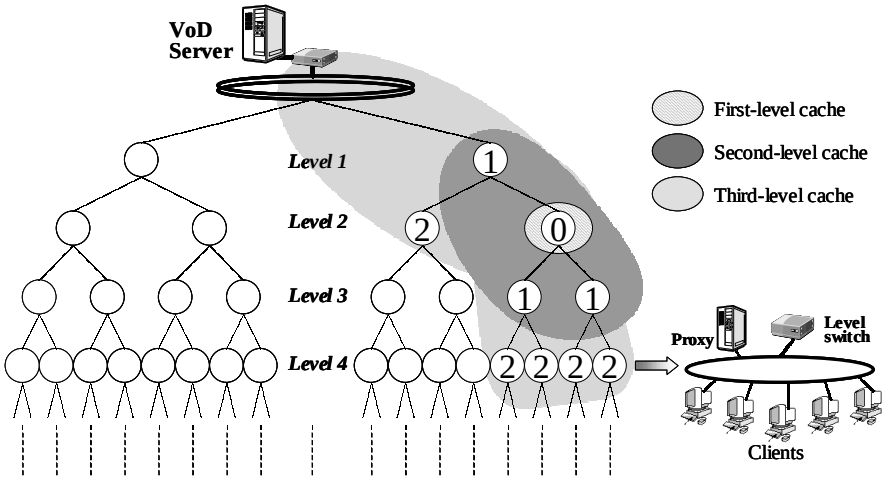

Fig. 2. Hierarchical proxy VoD system.

This hierarchical system has the advantage that, when one level's proxy misses the number of proxies that can be accessed increases too. That is, this topology allows the proxies group, formed by all those local networks situated at the same distance from the node in which the request has been made, to define a cache level in the topology hierarchy. The size of each cache level will be defined as the sum of the individual

capacity for each one of the proxies on this level. Also, figure 2 illustrates the different hierarchical levels of cache in the system.

On an initial analysis, a hierarchical tree topology has very desirable qualities from the perspectives of scalability and versatility; since the addition of a new local network only implies the inclusion of a switch to separate the traffic in the two network segments, without modifying any other component in the system. This topology also provides great versatility as the system can initially be composed of any number of networks, and can grow both in length (by adding new levels) and also in width (increasing the order of some tree nodes).

Nevertheless, the simple inclusion of a hierarchical system with proxies does not, in itself, obtain improvements in the system's scalability or efficiency: since, as all the proxies are replicating the same movies, if a proxy cannot serve a request from its client, then it is also very probable that none of the other proxies will be able to serve this request, and the solution will then require accessing the main server. We therefore need a new proxy organization and functionality that would allow us to increase the hit probability as the request climbs the various levels on the tree.

Consequently, our proposal lies in dividing the storage space available in proxies into two parts: one of these will continue functioning as a proxy, storing the most requested movies; the other proxy space will be used for making a distributed mirror of the main server's information. This new scheme is aimed at achieving:

1. To reduce the average distance of those requests that have failed in their local proxy, since, when distance is increased, mirror size also grows and the hit probability is greater.
2. To reduce server saturation, since all the requests generated in nodes situated beyond a determined server distance cannot reach the server, and therefore its workload is less.
3. To increase system fault tolerance, as the failure of a tree node does not prevent the rest of the system from continuing its work.

In order to evaluate both the classical system of proxies as well as our proposal based on the co-operation between cache and mirror schemes within the proxy, we are going to define an analytical model that allows us to compare the network efficiency and requisites offered by both approaches.

5 Analytical Model.

In order to validate the proposed architecture we have to demonstrate two important points: first, that this architecture can scale without causing network saturation (the main problem in scalability), and second, that the effectiveness lost by the scalability is not excessive. To do this we have developed an analytical model to measure the efficiency of the previously-mentioned proxy-based systems. In this model, we calculate the system's effective bandwidth that indicate the number of simultaneous streams that can be served simultaneously. In order to estimate the system's growth capacity, we also evaluate the growth of traffic generated by the system itself. These measurements will provide us with an idea of the system's limitation with respect to the number of users that it can admit, and its grade of scalability.

In order to realize this analysis, we assume a unicast policy, i.e., each user is assigned their own dedicated stream. This assumption is valid since our study is directed at evaluating the capacity of the system with respect to the independent streams that it is capable of managing. These results will be independent of whether bandwidth management policies can later be used (broadcasting, multicasting, etc.) in order to increase the efficiency and number of final clients for the system.

In addition, we assume a system with the following characteristics: a bandwidth for each local network of B_c Mbps, a proxy size that is sufficient to store Cp movies of the Vs available within the system, and a server and main network bandwidth of B_p Mbps.

5.1 One Level Proxy System.

As we have already commented, these systems are made up of a series of local networks, all of which are directly connected through the network to the main server.

Generically, the effective bandwidth of a proxy-based system (B_e) is evaluated as the maximum bandwidth available within the system (B_m) less the additional bandwidth (B_{fp}) required by proxy misses, given that these misses imply using the main network in order to attend them from the main server.

$$B_e = B_m - B_{fp} \quad \textbf{(1)}$$

This maximum bandwidth available within the system is obtained as the sum of the bandwidth of all the networks forming part of the SVoD, according to the following expression:

$$B_m = B_p + B_c \cdot n \quad \textbf{(2)}$$

where B_p is the bandwidth of the main network, B_c is the bandwidth of the local networks and *n* is the number of local networks.

In order to obtain the additional bandwidth (B_{fp}) required by proxy misses, we need to calculate the probability of these misses. If we assume that the proxies store the most watched Cp movies in their cache, and that the movie access pattern follows a Zipf(0) distribution with a skew factor of z, we can then calculate proxy-miss probability (p_{fp}) in accordance with the following formula:

$$p_{fp} = 1 - \sum_{m=1}^{Cp} \frac{1}{m^z \cdot \sum_{i=1}^{Vs} 1/i^z} \quad \textbf{(3)}$$

In this way, we can calculate additional bandwidth due to proxy miss as the probability of miss in each proxy (p_{fp}), multiplied by the traffic generated in all the system's local networks (B_c*n), that is:

$$B_{fp} = B_c \cdot n \cdot p_{fp} \quad \textbf{(4)}$$

This parameter indicates a bandwidth that the main network would be required to have in order for the system to be able to manage the *n* local networks.

Substituting expressions (2) and (4) within (1), effective bandwidth is then:

$$B_e = B_p + B_c \cdot n - B_c \cdot n \cdot p_{fp} = B_p + B_c \cdot n \cdot \left(1 - p_{fp}\right) \quad (5)$$

Using expressions (5) and (4), figure 3 illustrates a scalability analysis for one level proxy-based system, using effective bandwidth and the load received by the main network in accordance with the system's number of local networks. This study has been carried out for systems with different proxy sizes. The figure illustrates results using proxies with a capacity for storing 20%, 30% and 40% of the system's movies.

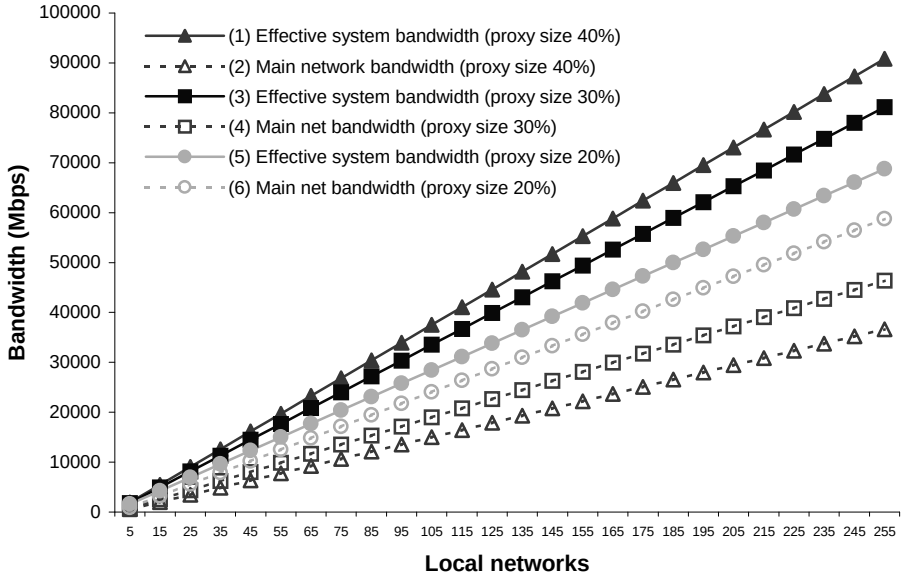

Fig. 3. Scalability of one level proxy based LVoD systems.

We can extract the following conclusions from the results shown in figure 3:

1. Proxy-based systems considerably increase the system's effective bandwidth (series 1, 3 and 5) and therefore the number of users that it can serve.
2. This system capacity increase is obtained at the expense of increasing the bandwidth requirements of the main network and main server. Thus, in series 2, 4 and 6, we can see that the width required by the main network increases linearly with the number of local networks within the system. Consequently, as the main network has a limited bandwidth (that cannot grow), the system has a limited capacity for growth.
3. Proxy cache size needs to have the capacity to store a high percentage of the movies available on the system. This is because, when a proxy is used with the capacity for 20% of the movies (series 5 and 6), the greater part of effective bandwidth obtained (68,000 Mbps for a system with 254 local networks) comes from the main network's bandwidth (58,000 Mbps), and therefore, for small proxy sizes, it is more economically viable to connect clients directly to the main network than to use local networks with proxies. This completely

centralized system would have an effective bandwidth of 58,000 Mbps, with a saving in costs for all the local networks.
Nevertheless, when proxy capacity is sufficiently large (sizes of 30% and 40%), the success in proxy cache increases, reducing access to the server network. For example, as we can see in figure 3, using a proxy with a size of 40% with 254 local networks (series 1 and 2), an effective bandwidth of 90,000 Mbps are obtained, which require nothing less than a main network with a 36,000 Mbps bandwidth.

5.2 Hierarchical Proxy-Based System.

Having seen the effectiveness of using one level proxy-based systems, we now consider in what ways we can extend this analysis in order to evaluate systems based on a hierarchical topology of proxies.

As a means of facilitating this study, we assume a complete tree topology (all levels are full), with L levels in which the order (o) identifies the number of local networks connected within each one of the tree nodes.

If we analyze expression (1) in the previous section, the only parameter that would be affected with the topology modification is additional bandwidth required, to serve client requests that have missed their local proxy. In this topology, when a request is not able to be met by the proxy of one given level, it accesses the following level, and so on, successively, increasing the bandwidth required. This cost (B_{fp}) will be determined by the number of levels (distance) that the request has cross before reaching the server or proxy that attends it, as we can see in figure 4.

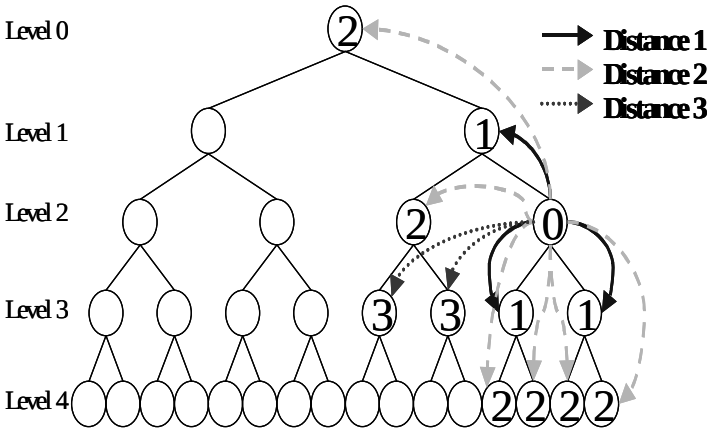

Fig. 4. Proxy misses distribution scheme.

5.3.1 Proxy with Caching.

As an initial approximation, in order to evaluate the total proxy-miss cost (B_{fp}) we could assume that cache misses can be obtained by adding the misses produced in every level. That is, the additional cost generated from a node of a given level can be

calculated as the percentage of its networks' traffic (B_c), which has failed in its proxy and which is being attended by proxies situated at distance 1 (B_c*p_{fp}), plus the percentage of traffic that is not served from the distance 1 proxies and that is attended by proxies at distance 2 (B_c*p_{fp}*p_{fp}), and so on, successively, until reaching level 0, in which the main server is found; this server will attend the request in the final instance. This cost, generated by a node of a given level, have to be multiplied by the number of nodes in this level in order to obtain the total bandwidth required by the level.

$$B_{fp} = B_c \cdot \sum_{v=1}^{L} o^v \sum_{d=1}^{v} p_{fp}^d \quad (6)$$

Nevertheless, this expression is not realistic as it assumes that the hit probability in higher-level proxies is the same as that for the proxy in its local network. Furthermore, as the proxies always attempt to store the most watched movies, a high level of information redundancy is generated, having negative repercussions on the proxy hit probability. This is because, if a request has already missed a proxy, it will very probably miss again with the remaining proxies, as they all replicate the same movies. It is therefore most likely that the hit probability in those proxies, situated at a distance greater than 1, from where the request has generated is practically zero.

A more realistic form of calculating additional cost in this hierarchical system is to assume that those requests that cannot be attended to by their local proxy have to be attended, instead, from the main server and - as a result - a cost proportional to distance (v), which separates them from the main server, will have to be considered. Taking this into account, total proxy-miss cost can be evaluated as:

$$B_{fp} \approx B_c \cdot \sum_{v=1}^{L} o^v \cdot p_{fp} \cdot v \quad (7)$$

This formula indicates that this hierarchical structure for proxies does not make sense, as it does not increase the probability of proxy success, and the distance that has to be covered by the requests, if the proxy fails, is greater, with the consequent increase in the penalization of the system's effective bandwidth.

We now move on to study the viability of this topology through the inclusion of a mirror in the proxies aimed at increasing the probability of their success.

5.3.2 Proxy with Caching and Mirroring.

With this new approach that we propose for proxies, their storage space is not modified, but rather is distributed between two different schemes: a percentage that continues managing the space as a cache for storing the most watched movies, and the rest that is used as a mirror of those movies less frequently accessed.

With this system, the probability of success from the initial node is greater, as the group of proxies situated at the same distance x from the local network in which the request is made can be seen as a single storage space formed by the group of mirrors from each of the proxies. In figure 4, we saw how a node situated at level 2, which misses its local proxy, can access 3 proxies situated at distance 1; if these are unable to attend the request, then it will have to access the 6 proxies situated at distance 2, and so on successively, until its request is attended by a given proxy, or it finally reaches its main server.

In order to facilitate this analysis, let us assume that the distribution of movies in each one of the mirrors is realized in an equal manner, and that this form of probability for success in the mirrors is constant and independent of its position in the hierarchy. From this supposition, the probable failure of a request made at level *v* and accessing all the proxies situated at a distance less than or equal to *d* is:

$$p_{fm}(v,d) = 1 - \left(p_{ap} + p_{fp} \cdot \frac{M_p \cdot Nd(v,d)}{V_s - C_p} \right) \quad (8)$$

in which:

p_{ap}	Is the probability of proxy cache hit (1-p_{fp}).
Nd(v,d)	Number of local networks at a distance less than or equal *d* from level *v*.
Mp	Is the proxy space reserved for the mirror movies.

This expression indicates that the probability of success at distance 0 (local proxy in which the request was made) is the accumulated probability that the request can be met from cache or from the proxy mirror, whilst, for greater distances, only the probability of success for proxy mirrors situated at the same distance is considered.

In this way, the required bandwidth for proxy misses (cache+mirror) would be evaluated as:

$$B_{fp} = B_c \cdot \sum_{v=1}^{L} o^v \cdot \sum_{d=0}^{v} p_{fm}(v,d) \quad (9)$$

Substituting this expression in the expression (1), we obtain the following effective bandwidth:

$$B_e = B_m - B_c \cdot \sum_{v=1}^{L} o^v \cdot \sum_{d=0}^{v} p_{fm}(v,d) \quad (10)$$

On the other hand, we can evaluate the traffic (requests) received by the main server, which characterizes overload in the main network as:

$$S_p = B_c \cdot \sum_{v=0}^{L} \frac{o^v \cdot p_{fm}(v,v)}{r_s(v)} \quad (11)$$

This expression implies that, as the distance grows between the networks in which the requests were generated and the server, the number of proxies that can attend them also grows, thereby reducing the volume of traffic that arrives at the main network. The parameter r_s (request to server) identifies the traffic percentage that is attended by the main server. If the traffic generated by proxy misses is distributed equally between neighboring mirrors situated at the same distances then r_s can be evaluated as:

$$r_s(v) = Nd(v,v) \quad (12)$$

But, the request cannot always be distributed equally between proxies and the main server, due to proxies having only a portion of the movies and not being able to serve as many requests as the main server. In order to avoid this assumption we propose another distribution based on hit probability, as we show in the following expression:

$$r_s(v) = 1 + \frac{p_{fm}(v,v) - p_{fm}(v,v+1)}{p_{fm}(v,v)} \quad (13)$$

Summarizing the proposed scheme, based on proxies with caching and mirroring, we can say that it behaves as a VoD distributed architecture, as it manages to decentralize both the traffic from the communications system (through the hierarchical system) as well as that of the server itself (through the mirrors and caches).

6 Evaluation of the Proposed Architecture.

In this section, we will use the previously-defined analytical model in order to evaluate the scalability and performance of the proposed topology.

As we have commented, in order to demonstrate that our topology is scalable, we need to ensure that system growth does not modify the requirements (bandwidth) of the existing elements, or, if it grows, that this growth is limited. To do this, we have to make sure that none of the bandwidths in any of the system components grows when the system expands. In hierarchical systems, as the element that can receive greatest load is that situated at the highest level of the hierarchy (the server and the main network in our case), then we only need to prove that the main network and the server do not become saturated as the system grows.

We assume that the system used has the following characteristics: the bandwidth of the main and local networks is 500 Mbps, it has 100 different movies and an access pattern modeled through a Zipf(0) distribution, with a skew factor of 0.729 (the asymmetry factor that is most used in the literature for modeling VoD systems [1]). Figure 5 illustrates the effective bandwidth (B_e) according to the expression (10) and the main network load (S_p) defined by expression (11), for LVoD systems based on topologies with binary trees, and using proxies with a capacity for 30% and 40% of the movies.

As can be observed, as the number of connected local networks grows, main network load not only grows but also initially diminishes, only to stabilize itself later (at around 280 Mbps with proxies of a 40% size and at 380 Mbps with a 30% size, series 3 and 4 respectively), independently of the number of levels that are added. Therefore, having assumed in the analytical model a homogenized behavior for all networks (this means that clients are also connected to the main network), the previously-obtained results can be directly extrapolated to the other levels, and can therefore guarantee the scalability of all the system networks.

Moreover, the performance obtained for this system, understood as its effective bandwidth (series 1 and 2), grow linearly as the number of local networks increases. If we assume that, in the classical system of one level proxy, the system's main network always has sufficient capacity to attend to all proxy misses, and that therefore

the bandwidth of this network grows on the system's expansion, we can compare the performance obtained by our architecture with those obtained through a system of one level proxy (Fig. 3).

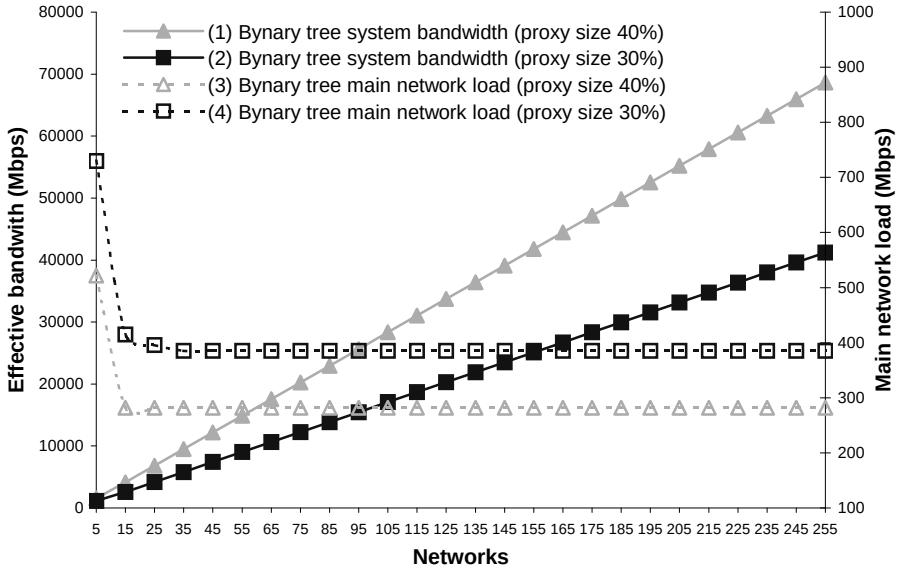

Fig. 5. Scalability of a hierarchical proxy LVoD system with caching and mirroring.

From such a comparison, we can conclude that a decrease in the effective bandwidth of a hierarchical system with respect to the same system, but with a proxy level, depends on the size of the proxies in question. That is, this loss of performance can be quantified at 49% (81000 Mbps in the figure 3 as against 41000 Mbps in figure 5) for a system with 254 local networks and a 30% proxy. Nevertheless, if we increase proxy size to 40% of the system's movies, then the difference in performance hardly reaches 23% (90000 Mbps as against 69000 Mbps). This reduction in performance is due to the fact that when sufficiently large proxies are used, misses do not extend beyond those proxies situated at distance 1 (exactly as occurs in one-level proxy system), improving performance.

On the other hand, performance in this hierarchical proxy system can also be affected by the distribution of proxy storage space between caching and mirroring schemes. In order to study the influence of this parameter, in the figure 6 we show the performance obtained by the system whilst the Cp parameter has been varied in expression (8), which measures the proxy-hit probability. The results have been obtained using a system formed by 254 local networks (7 levels) and with a proxy capacity to store 30% and 40% of the movies. The figure illustrates the effective bandwidth as well as maximum, mean and minimum bandwidth (main network) for system networks.

As we can see, when we use a lower capacity proxy (30%, fig 6a), the best result is obtained using a mirror with 73% of the available space, leaving the cache with the 27% remaining. On using a larger proxy (40%, fig 6b), the percentages vary, the best

performance being obtained when proxy space is distributed equally between the two schemes.

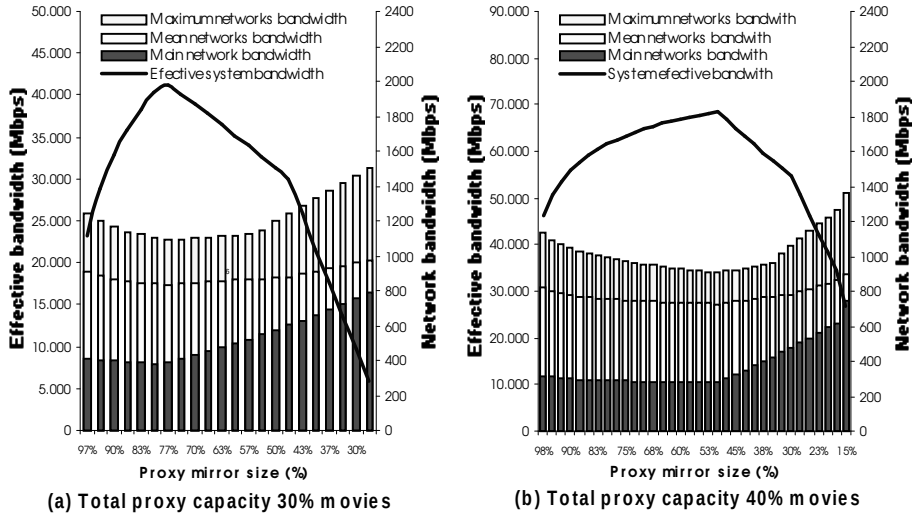

Fig. 6. Cache and mirror size with a binary tree topology.

7 Conclusions.

We have proposed a hierarchical tree topology based on the use of independent local networks. These networks use a local proxy in order to meet requests without the need to access the main server. In case of proxy miss, the server is not accessed directly, but instead, the system attempts to server the request from closer local networks within the topology. In order to increase system scalability, the functionality of the proxy has been modified in such a way that it works at the same time as cache for the most watched movies, and as a mirror for the remaining movies.

This architecture guarantees unlimited and low-cost growth for the VoD system. Additionally, the capacity of the system can be easily adapted to any number of users, and therefore does not require a initial over-dimensioning in order to facilitate subsequent growth. This increase in system capacity may facilitate the inclusion of new services (video conference, access to high quality multimedia Internet contents, etc.) in order to increase system profitability.

The use of a distributed architecture results in a system that is fault-tolerant. And the use of independent networks and distributed mirrors in the proxies to replicate the system's movies allows for the fact that, although any of the networks, proxies or even the main server might fail, service can partially be continued from the local network proxies. Furthermore, by distributing the request service between proxies, the size of the service required (and its cost) is smaller that of a partially or totally centralized system.

Although the bandwidth required by the tree topology is greater, this does not necessarily imply greater cost. One-level proxy systems require a main network with a bandwidth that is considerably greater than that of a hierarchical system. Assuming that the cost of a network does not grow in a linearly with the bandwidth, but rather, that its growth is more likely to be exponential, we can then predict that the cost of the tree topology will be lower.

On the other hand, the architecture proposed (in keeping with most distributed systems) needs to sacrifice part of its efficiency in order to withstand fault tolerance and scalability. Nevertheless, as we have confirmed, this reduction in features decreases with the increase in proxy storage capacity.

Our future research will focus on increasing hierarchical topology performance through an increase in proxy-hit probability and by adapting classical bandwidth management policies (broadcasting, multicasting) to our architecture.

8 Bibliography.

1. A. Dan, D. Sitaram, and P. Shahabuddin, "Dynamic batching policies for an on-demand video server," Multimedia Systems 4, pp. 112--121, June 1996.
2. A. Dan, D. Sitaram, and P. Shahabuddin, “Scheduling Policies for an On-Demand Video Server with Batching”, *Proc. 2nd ACM Int’l. Multimedia Conference (ACM MULTIMEDIA ’94)*, San Francisco, CA, Oct. 1994, pp. 15-23.
3. C. Aggarwal, J. Wolf, and P. S. Yu, "On optimal piggybacking merging policies for video-on-demand systems," Performance Evaluation Review, vol. 24, pp. 200--209, May 1996.
4. D. L. Eager, M. K. Vernon, and J. Zahorjan, "Minimizing Bandwidth Requirements for On-Demand Data Delivery", Proc. MIS'99, Indian Wells, CA, Oct. 1999.
5. F. A. Tobagi, “Distance learning with digital video,” IEEE Multimedia Magazine, pp. 90-94, Spring 1995.
6. K. A. Hua and S. Sheu, "Skyscraper Broadcasting: a new broadcasting scheme for metropolitan video-on-demand systems," in SIGCOMM 97, pp. 89--100, ACM, (Cannes, France), Sept. 1997.
7. Hua, Ying Cai and Simon Sheu, Patching : A Multicast Tecnique for true Video-on-Demand Services, ACM Multimedia’98, pages 191-200.
8. Luigi Rizzo and Lorenzo Vicisano. “Replacement policies for a proxy cache.” Technical Report RN/98/13, UCL-CS, 1998.
9. S. Sen, L. Gao, J. Rexford, and D. Towsley, "Optimal Patching Schemes for Efficient Multimedia Streaming", Proc. 9 th Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'99), Basking Ridge, NJ, June 1999.
10. S. Sheu, K. A. Hua, and W. Tavanapong “Chaining: A Generalized Batching Technique for Video-On-Demand Systems”, In Proc. IEEE Int’l Conf. On Multimedia Compunting and Systems (ICMCS)’97, Ottawa, Ontario, Canada, June 3-6, 1997, pp. 110-117.
11. S. Viswanathan and T. Imielinski, "Metropolitan area video-on-demand service using pyramid broadcasting," Multimedia Systems 4, pp. 197--208, Aug. 1996.
12. S.-H. G. Chan and F. Tobagi, “Caching schemes for distributed video services”, in Proceedings of the 1999 IEEE International Conference on Communications (ICC'99), (Vancouver, Canada), June 1999.
13. S.W. Lau, J.C.S. Lui and L. Golubchik, Merging Video Streams in a Multimedia Storage Server: Complexity and Heuristics, Multimedia Systems, 6(1), 1998, 29-42.
14. Subhabrata Sen, Jennifer Rexford, and Don Towsley. “Proxy prefix caching for multimedia streams.” In Proceedings of the IEEE Infocom, 1999.